Chapter 1
An SL/QP Algorithm for Minimizing the Spectral Abscissa of Time Delay Systems

Vyacheslav Kungurtsev*, Wim Michiels **, and Moritz Diehl ***

We consider a problem of eigenvalue optimization, in particular finding a local minimizer of the spectral abscissa - the value of a parameter that results in the smallest value of the largest real part of the spectrum of a system. This is an important problem for the stabilization of control systems but it is difficult to solve because the underlying objective function is typically nonconvex, nonsmooth, and non-Lipschitz. We present an expanded sequential linear and quadratic programming algorithm that solves a series of linear or quadratic subproblems formed by linearizing, with respect to the parameters, a set of right-most eigenvalues at each point as well as historical information at nearby points. We present a comparison of the performance of this algorithm with the state of the art in the field.

1.1 Introduction

We are interested in optimizing the spectrum of continuous time systems. Recall that finding the spectrum of a time-delay system of the form,

* Agent Technology Center, Department of Computer Science, Faculty of Electrical Engineering, Czech Technical University in Prague. Research supported by Cisco-Czech Technical University Sponsored Research Agreement project WP5 (vyacheslav.kungurtsev@fel.cvut.cz).

** Computer Science Department and Optimization in Engineering Center (OPTEC), KU Leuven, Kasteelpark Arenberg 10, B-3001 Leuven- Heverlee, Belgium. (wim.michiels@cs.kuleuven.be).

*** Department of Microsystems Engineering IMTEK, University of Freiburg, Georges-Koehler-Allee 102, 79110 Freiburg, Germany (moritz.diehl@imtek.uni-freiburg.de).

$$v'(t) = \sum_{j=0}^{m} A_j(x) v(t - \tau_j).$$

presents a nonlinear eigenvalue problem. We will assume that $\tau_0 = 0$. To solve for the eigenvalues, we find the solutions $\lambda(x)$ of,

$$\det(\Lambda(\lambda; x)) = 0,$$

with,

$$\Lambda(\lambda; x) = \lambda I - A_0(x) - \sum_{j=1}^{m} A_j(x) e^{-\lambda \tau_j}.$$

The number of eigenvalues in this case is generally infinite, but within any right half-plane the number of eigenvalues is finite [17]. We let $F(x)$ be the infinitesimal generator corresponding to the solution operator of the delay system, and the spectrum as $\sigma(F(x))$.

As an illustrative special case, consider a simpler problem of optimizing the spectrum of a linear system controlled with static, undelayed output feedback, where the operator $F(x)$ reduces to the matrix,

$$F(x) = A + BXC,$$

where A is the open-loop matrix for the system, B the input matrix and C the output matrix, and X is formed by arranging the components of x into a matrix of the appropriate dimensions. This presents a linear eigenvalue problem, with a finite spectrum.

The problem of interest can be written in the form,

$$\min_{x \in \mathbb{R}^n} f(x) = \min_{x \in \mathbb{R}^n} \alpha(F(x)), \tag{1.1}$$

where the spectral abscissa α is defined to be,

$$\alpha(F(x)) = \max_{\lambda \in \sigma(F(x))} \mathfrak{Re}\lambda.$$

The spectral abscissa corresponds to the largest real part of the eigenvalues of $F(x)$.

The properties of the spectrum of a matrix subject to parameters is an involved topic, for an early work, see [1]. A more thorough analysis with respect to the spectral abscissa in particular was presented in [4]. For recent work see, for instance [5, 14]. An important fact that permits a lot of the subsequent analysis is that the spectrum $\{\lambda_0(x), \lambda_1(x), ..., \lambda_{N-1}(x)\}$ of a matrix $F(x)$ is a continuous function of x. Typically, local minimizers correspond to points x at which some of the eigenvalues coalesce, i.e., $\mathfrak{Re}(\lambda_0(x)) = \mathfrak{Re}(\lambda_1(x)) = ...$ In [22] it was shown, however, that although for symmetric $F(x)$, $f(x)$ is convex, in the nonsymmetric case $f(x)$ is not even Lipschitz. If for all x, all of the

active eigenvalues (i.e., λ_i such that $\mathfrak{Re}(\lambda_0(x)) = \mathfrak{Re}(\lambda_i(x))$ were simple, then $f(x)$ would correspond to the maximum of a set of smooth surfaces. However, this is typically not the case. Thus, the optimization problem is difficult to solve because it is nonconvex, nonsmooth, and typically non-Lipschitz.

It can be observed, however, that the extensive variational analysis of the spectral abscissa has been performed for matrix eigenvalue optimization, rather than time delays. The main difference lies in the fact that in the generic case there are infinitely many eigenvalues. Thus, at this point, we can expect that optimizing the spectrum of a nonlinear eigenvalue problem should be *at least as difficult* as for matrices, and so all of the variational properties presenting challenges extend appropriately.

We present the plot of a two-dimensional problem in Figure 1.1. In this example, first given in [23], $F(x) = A + BK$, with,

$$A = \begin{pmatrix} 0.1 & -0.03 & 0.2 \\ 0.2 & 0.05 & 0.01 \\ -0.06 & 0.2 & 0.07 \end{pmatrix}, \; B = \frac{1}{2}\begin{pmatrix} -1 \\ -2 \\ 1 \end{pmatrix}, \; K^T = \begin{pmatrix} x_1 \\ x_2 \\ 1.4 \end{pmatrix}$$

Notice that all of the features of $\alpha(F(x))$ we describe above, nonconvexity, nonsmoothness and non-Lipchitz behavior are evident in the figure.

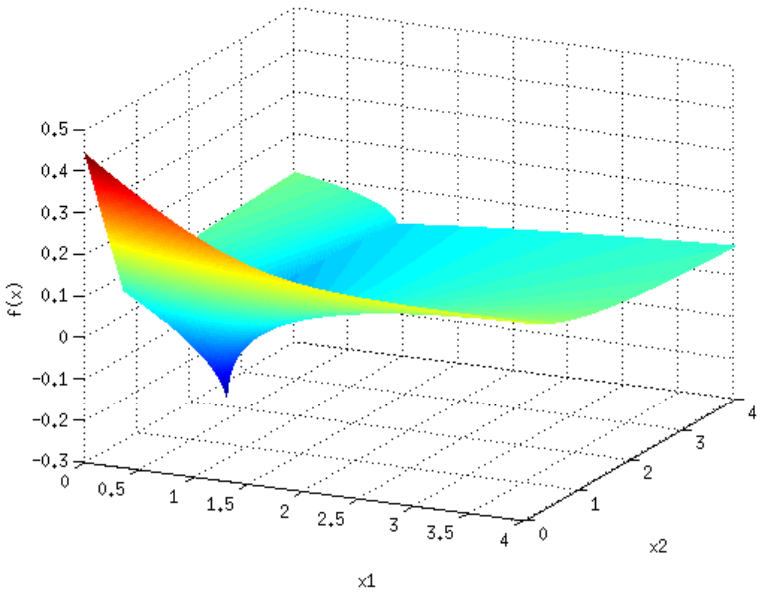

Fig. 1.1: $\alpha(F(x))$ for a two-dimensional eigenvalue optimization problem.

1.2 Algorithms for Eigenvalue Optimization

1.2.1 State of the Art

In sampling methods, a set of gradients is generated by sampling around the current point, which serves to approximate the Clarke subdifferential. At each iteration, a step in the convex hull of the sampled gradients is taken [2, 3]. The authors test the algorithm for non-delayed problems in eigenvalue optimization. In addition, variable metric methods, in particular of the Broyden-Fletcher-Goldfarb-Shanno (BFGS) form, have been shown to exhibit good convergence properties for nonsmooth problems, albeit without convergence theory [15]. A variant BFGS-gradient sampling hybrid, in which sampling is performed within the quasi-newton updates, has also been shown to globally converge [7] using a dual set of QPs to generate the step from a sample of gradients. It should be noted, however, that the proven convergence of gradient sampling relies on the objective function being locally Lipschitz, which is not the case for the spectral abscissa in particular.

In bundle methods, originally developed for convex nonsmooth optimization build gradient information by maintaining historically calculated subgradients during the course of the iterations, and at each point solve cutting-plane underapproximations of the function [11, 13]. In the case of non-convex problems, the formulation and analysis of bundle methods is more complicated, but various variations of bundle algorithms exist (see, e.g., [9], and a nice survey in the beginning of [8]).

In application to eigenvalue optimization, for non-delayed matrices, variations of the bundle method in [18, 19] consider a method wherein the ϵ Clarke-subdifferential is approximated by incorporating eigenvalues in the spectrum that are within ϵ of the abscissa and including their derivatives with respect to x in the subproblem. This includes thorough and interesting convergence theory, however only symmetric matrices are considered.

In [24] gradient sampling is applied for optimizing the spectral abscissa of a time-delay system. The procedure was found to be robust in terms of generating a sequence of directions of decrease until a stationary point of the spectral abscissa function is found.

1.2.2 Sequential Linear and Quadratic Methods

1.2.2.1 Basic Algorithm

It is possible to formulate a sequential quadratic programming (SQP) method, as well as a more simplified Sequential Linear Programming (SLP) method for minimizing the spectral abscissa. First presented for symmetric matri-

ces in [20, 21], this approach is based on the realization that the problem $\min_{x\in\mathbb{R}^n} \alpha(F(x))$ can be rewritten as,

$$\begin{aligned} &\min_{\gamma\in\mathbb{R},x\in\mathbb{R}^n}\ \gamma, \\ &\quad \text{subject to } \gamma \geq \mathfrak{Re}(\lambda_i(F(x))) \text{ for all } i. \end{aligned} \tag{1.2}$$

This is a standard formulation for max-min programming. However, if we are to consider this problem in its entirety, we would have a semi-infinite programming problem, because in general a time-delay system will have an infinite number of eigenvalues. However, in any right half plane, the number of eigenvalues is finite, and so since we are interested in minimizing the rightmost eigenvalue, it is natural to consider the problem, instead,

$$\begin{aligned} &\min_{\gamma\in\mathbb{R},x\in\mathbb{R}^n}\ \gamma, \\ &\quad \text{subject to } \gamma \geq \mathfrak{Re}(\lambda_i(F(x))) \text{ for all } i \text{ such that } \mathfrak{Re}\lambda_i(F(x)) > \lambda_c, \end{aligned}$$

where λ_c separated the plane that we restrict our attention to, and will depend on the number of eigenvalue surfaces we want/are able to incorporate and the location of these eigenvalues. We will order the spectrum as $\mathfrak{Re}(\lambda_0(F(x))) \geq \mathfrak{Re}(\lambda_1(F(x))) \geq \mathfrak{Re}(\lambda_2(F(x))) \geq ... \geq \mathfrak{Re}(\lambda_{N_c}(F(x)))$ where $N_c + 1$ is the number of eigenvalues that fall to the right of λ_c.

In the case that the eigenvalues of $F(x)$ are isolated and simple, the gradient and Hessian of $\lambda_i(F(x))$ with respect to x is well defined for all $\lambda_i(F(x)) \in \sigma(F(x))$. In this case, the objective function is the maximum of a set of smooth surfaces, and is thus piece-wise smooth. Solving the problem by successive approximation of each surface is standard, and the associated convergence theory in [20,21] proves that the procedure outlined below converges to the solution.

Of course, in the general setting, eigenvalues need not be simple and isolated. The number of points at which the objective function $\alpha(F(x))$ in (1.1) is nonsmooth is of measure zero in the Lebesgue space $\mathbb{R}^n$. This implies that for a.e. x, the function $\alpha(F(x))$ is a locally smooth surface. This surface corresponds to the value of $\lambda_0(F(x))$ as a function of x. This implies that the algorithm (and computing linearizations) is well-defined at every point. However, as minimizers tend to be points of nonsmooth and non-Lipschitz behavior, such a scheme is no longer certifiably convergent. In practice, however, following some modifications, we will see that it still performs well.

For now, consider the simple case that all eigenvalues $\lambda_i(F(x))$ are simple and isolated. It can be shown that, in the case where each $A_i(x)$ depends smoothly on x and where the eigenvalue has multiplicity 1, the derivative of the surface corresponding to each eigenvalue as well as $\nabla^2_{xx}\lambda_i(F(x))$ can be calculated from the formulas [12, 16, 17].

$$\nabla_x \lambda_i = \frac{u_i^* \left(\frac{\partial A_0}{\partial x} + \sum_{j=1}^m \frac{\partial A_j}{\partial x} e^{-\lambda_i \tau_j} \right) v_i}{u_i^* \left(I + \sum_{j=1}^m \tau_j e^{-\lambda \tau_j} A_j \right) v_i},$$

where u_i and v_i are the left and right eigenvectors of $F(x)$ corresponding to eigenvalue i, u^* corresponds to the conjugate of u and the second-derivatives may be calculated explicitly by

$$\nabla^2_{xx}\lambda_i(x) = -\frac{u_i^*(\nabla^2_{x\lambda}\Lambda(\lambda_i,x)\otimes\nabla_x\lambda_i+\nabla^2_{xx}\Lambda(\lambda_i,x)+\nabla^2_{\lambda\lambda}\Lambda(\lambda_i,x)\otimes(\nabla_x\lambda_i)\otimes(\nabla_x\lambda_i))v_i}{u_i^*\nabla_\lambda\Lambda(\lambda_i,x)v_i} + \frac{u_i^*(2\nabla_x\Lambda(\lambda_i,x)+2\nabla_\lambda\Lambda(\lambda_i,x)\otimes\nabla_x\lambda_i)\nabla_x v_i}{u_i^*\nabla_\lambda\Lambda(\lambda_i,x)v_i},$$

where $\nabla_x v_i$ can be calculated (along with $\nabla_x\lambda_i$) by,

$$\begin{pmatrix}\Lambda(\lambda_i,x) & \nabla_\lambda\Lambda(\lambda_i,x)\\ 2v_i^* & 0\end{pmatrix}\begin{pmatrix}\nabla_x v_i\\ \nabla_x\lambda_i\end{pmatrix} = \begin{pmatrix}\nabla_x v_i\\ 0\end{pmatrix},$$

where the second set of equations comes from differentiating $v_i^* v_i = 1$.

The Lagrangian function for the problem (1.2) is defined as,

$$L(\gamma, x, y) = \gamma - \sum_{i=0}^{N_c} y_i(\gamma - \mathfrak{Re}\,(\lambda_i(F(x)))), \tag{1.3}$$

where y is the vector of Lagrange multipliers.

This naturally suggests the SQP method wherein a sequence of iterations $x_{k+1} = x_k + t\Delta x$ is calculated, with t a line-search scalar and Δx is determined by solving subproblems of the form,

$$\begin{aligned}&\min_{\Delta x,\Delta\gamma}\ \Delta\gamma + \tfrac{1}{2}\Delta x^T H_k \Delta x,\\ &\text{subject to } \Delta\gamma + \alpha(F(x_k)) \geq \mathfrak{Re}(\lambda_i(F(x_k))) + \mathfrak{Re}(\nabla_x\lambda_i(F(x_k)))^T\Delta x,\\ &\qquad\qquad \forall i \in \{0,...,N_c\},\end{aligned} \tag{1.4}$$

for Δx and $\Delta\gamma$, where H_k is a Lagrangian Hessian term at x_k.

The nonconvexity of the problem implies that at any local quadratic approximation of an eigenvalue surface, the Hessian could be indefinite or even negative definite. This implies that the approximating quadratic program (1.4) could be unbounded below. We constrain the problem with a trust-region to prevent this. Since we have linear constraints, making an $l2$ norm constraint impractical, we use an infinity norm trust-region, which acts as a "box" limiting the magnitude of the maximal component of Δx.

$$\begin{aligned}&\min_{\Delta x,\Delta\gamma}\ \Delta\gamma + \tfrac{1}{2}\Delta x^T H_k \Delta x,\\ &\text{subject to } \Delta\gamma + \alpha(F(x_k)) \geq \mathfrak{Re}(\lambda_i(F(x_k))) + \mathfrak{Re}(\nabla_x\lambda_i(F(x_k)))^T\Delta x,\ \forall i,\\ &\qquad\qquad \forall i \in \{0,...,N_c\},\\ &\qquad ||\Delta x||_\infty \leq \Delta_k.\end{aligned} \tag{1.5}$$

Since the Hessian could be indefinite, the solution Δx could be a direction of ascent for the objective function. Hence, after computing Δx we first test if,

$$\alpha(F(x_k + \Delta x)) < \alpha(F(x_k)), \tag{1.6}$$

in which case we set $x_{k+1} = x_k + \Delta x$ and continue to the next iteration. Otherwise, we test for descent,

$$\mathfrak{Re}(\nabla_x \lambda_i(F(x_k)))^T \Delta x < 0, \tag{1.7}$$

and if this does not hold we set $\Delta_{k+1} = \gamma_1 \Delta_k$, where γ_1 is a constant satisfying $\gamma_1 \in (0, 1)$, and resolve the subproblem.

If (1.7) holds, we follow the mixed trust-region/line-search procedure presented by Gertz [10], in which a backtracking line search reduces the size of the step t until decrease is achieved ($\alpha(F(x_k + t\Delta x)) < \alpha(F(x_k))$), and the next trust-region radius corresponds to $t||\Delta x||$.

$$\Delta_{k+1} = \begin{cases} \gamma_2 \Delta_k & \text{if } \alpha(F((x_k + \Delta x)) < \alpha(F(x_k)) \\ t||\Delta x|| & \text{otherwise,} \end{cases} \tag{1.8}$$

where γ_2 is a constant satisfying $\gamma > 1$.

We update the trust-region simply by increasing it if we achieve descent, and decreasing it otherwise. For consistency with convergence theory [6], we would enforce sufficient decrease conditions with respect to predicted (from the quadratic approximation) and actual decrease. However, since lax criteria of acceptance (e.g., with a small constant multiplying the predicted-actual decrease ratio) of the step is practically equivalent to this condition, we proceed as in the line-search criteria for the gradient sampling method [3] to just enforce descent.

If we omit the second order term H_k, then the algorithm becomes a sequential linear programming (SLP) method. With the trust-region, the solution is always bounded.

1.2.2.2 Incorporating historical gradients

We found that in the nonsymmetric case, the basic SL/QP algorithm would frequently stall at non-optimal points. Recall that the reliability of the algorithm depended on some strong assumptions on the problem. To give a generic geometric picture of the situation for which this occurs, consider a "valley", or $n-1$ dimensional hypersurface in $\mathbb{R}^n$ at which $\nabla \lambda_i(F(x))$ is undefined. It can happen that across the $n-1$ dimensional manifold of x on which this occurs, the derivatives of $\lambda_i(F(x))$ jump discontinuously.

Locally, the directional derivative of $\alpha(F(x))$ is steeper towards the valley than parallel along it, so a local approximation that regards only the eigenvalue surfaces at a point on one side of the valley will result in the step of steepest decrease being in this direction. Since the surface on the other side of the valley is not accounted for on the original side, this is not incorporated directly into the subproblem. We illustrate this scenario in Figure 1.2.

To remedy this, we added "memory" to the SQP method with a set $\mathcal{M}$,. Essentially this behaves as a bundle, whose elements (eigenvalue value and derivative with respect to x) are included alongside the linearizations of the rightmost eigenvalues at the current point. When it occurs that $\alpha(F(x_k + \Delta x)) > \alpha(F(x_k))$, which we expect in the "jamming" scenario described above, the procedure stores the tuple $\{x_k + \Delta x_k, \mathfrak{Re}(\lambda_0(F(x_k + \Delta x_k))), \nabla_x \mathfrak{Re}(\lambda_0(F(x_k+\Delta x_k)))\}$ in $\mathcal{M}$. Then, if in a future iteration K, the current point x_K satisfies $||x_K - x^{(i)}||_\infty \leq \Delta_k$ for any $i \in \{1, ..., |\mathcal{M}|\}$, then we include this linearized surface in the QP subproblem. Thus at each iteration, subproblem (1.9) is solved to generate the trial point for the linar search.

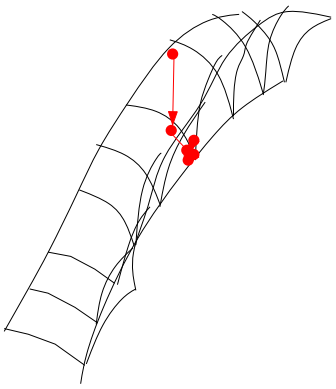

Fig. 1.2: Possible set of iterations of SQP without memory along a surface of $\alpha(F(x))$.

$$
\begin{aligned}
&\min_{\Delta x, \Delta\gamma} \ \Delta\gamma + \tfrac{1}{2}\Delta x^T H_k \Delta x, \\
&\text{subject to } \Delta\gamma + \alpha(F(x_k)) \geq \mathfrak{Re}(\lambda_i(F(x_k))) + \mathfrak{Re}(\nabla_x \lambda_i(F(x_k)))^T \Delta x, \ \forall i \\
&\qquad \Delta\gamma + \alpha(F(x_k)) \geq \mathfrak{Re}(\lambda_{(i)}(F(x^{(i)}))) \\
&\qquad\qquad + \mathfrak{Re}(\nabla_x \lambda_{(i)}(F(x^{(i)})))^T (x_k + \Delta x - x^{(i)}), \ i \in M_k \\
&\qquad ||\Delta x||_\infty \leq \Delta_k,
\end{aligned}
\tag{1.9}
$$

where $M_k \subset \mathcal{M}$ represents the points $x^{(i)}$ satisfying $||x^{(i)} - x_k|| \leq \Delta_k$.

We found in our experiments that this way of incorporating historical information proved to be the most effective. In genereal, on the smooth portions of the spectral abscissa function, the standard SLP/SQP produces iterates that quickly converge towards a region with a lower objective, then memory is needed to refine the solution to find a more precise local minimizer.

1.2.2.3 Algorithm Summary

We present a summary of the procedure, the SLP variation, in Algorithm 1 Note that we can select a subset of the eigenvalues N_k at each iteration k to evaluate and linearize. The stopping criterion corresponds to the step becoming small, without any new information (memory) being added at the current iteration.

Algorithm 1 SLP Algorithm for Eigenvalue Optimization.

Define constants $0 < \gamma_1 < 1$, $\gamma_2 > 1$, $\delta_m > 0$, and $S \in \mathbb{N}$.
Determine N_0 or λ_c.
for S times **do**
  Randomly select starting point x_0.
  Set $\mathcal{M}_1 = \emptyset$. Set $k = 1$.
  Calculate initial $\{\lambda_i(F(x_0))\}$ and $\{\nabla_x \lambda_i(F(x_0))\}$ for $i \in \{0, .., N_0\}$.
  while ($||\Delta x|| > \delta_m$ or $\mathcal{M}_k \neq \mathcal{M}_{k-1}$) **do**
    Solve

$$\begin{aligned} &\min_{\Delta x, \Delta\gamma} \ \Delta\gamma, \\ &\text{subject to } \Delta\gamma + \alpha(F(x_k)) \geq \mathfrak{Re}(\lambda_i(F(x_k))) \\ &\qquad\qquad +\mathfrak{Re}(\nabla_x \lambda_i(F(x_k)))^T \Delta x, \ i \in \{0, ..., N_k\} \\ &\qquad \Delta\gamma + \alpha(F(x_k)) \geq \mathfrak{Re}(\lambda_{(i)}(F(x^{(i)}))) \\ &\qquad\qquad +\mathfrak{Re}(\nabla_x \lambda_{(i)}(F(x^{(i)})))^T (x_k + \Delta x - x^{(i)}), \ i \in M_k \\ &\qquad ||\Delta x||_\infty \leq \Delta_k. \end{aligned} \tag{1.10}$$

    for Δx_k.
    Calculate $\{\lambda_i(F(x_k + \Delta x_k))\}$ and $\{\nabla_x \lambda_i(F(x_k + \Delta x_k))\}$ for $i \in \{0, .., N_k\}$
    if $\alpha(F(x_k + \Delta x_k)) < \alpha(F(x_k))$ **then**
      Set $x_{k+1} \leftarrow x_k$
      Set $\Delta_{k+1} \leftarrow \gamma \Delta_k$.
    else
      Store $\{x_k + \Delta x_k, \mathfrak{Re}(\lambda_0(F(x_k + \Delta x_k))), \nabla_x \mathfrak{Re}(\lambda_0(F(x_k + \Delta x_k)))\}$
        in $\mathcal{M}_{k+1}$.
      Find t such that $\alpha(F(x_k + t\Delta x_k)) < \alpha(F(x_k))$.
      Set $x_{k+1} \leftarrow x_k + t\Delta x_k$.
      Set $\Delta_{k+1} \leftarrow t||\Delta x_k||$.
    end if
    Set $k \leftarrow k + 1$.
    Determine N_k. Typically, set $N_k = N$, the size of $F(x)$.
    Calculate all $\{\lambda_i(F(x_k))\}$ and $\{\nabla_x \lambda_i(F(x_k))\}$ for $i \in \{0, .., N_0\}$.
  end while
  Add the last point $(x_f, \alpha(F(x_f)))$ to $\mathcal{F}$.
end for
  return $\{x_f, \alpha(F(x_f))\}$ corresponding to the lowest value of $\alpha(F(x_f))$ in $\mathcal{F}$.

Finally, we note that since this problem is nonconvex, there can be multiple local minima, possibly necessitating the use of global optimization strategies. In our experiments we have found this to be problem dependent, i.e., for some systems there are many local minima, but not others. Given that the objective function is not a closed form function, determinstic strategies for global optimization would be impossible to implement. In our implementation, we use ten random starting points, initialized as a random normal variable centered at zero, and select the lowest minimizer out of the ten runs. In practice, for many problems, it is expected that the parameters should lie in some bounded region, permitting the use of more probabilistically sophisticated strategies [25]. In addition, in many applications, only a point at which the system is stable, e.g., the spectral abscissa is below zero, is needed

rather than the absolute global minimizer, and so there would be some laxity in the treatment of the existence of multiple local minimizers.

1.3 Numerical Results

For all solvers, we used a stopping tolerance of 1e-4, indicating that the algorithms terminate when the (inf) norm of the previous step was smaller than 1e-4. The SL/QP algorithms were coded in MATLAB, with all tests run using MATLAB version 2013a and were performed on an Intel Core 2.2 GHz $\times 8$ running Ubuntu 14.04. For all algorithms we use the same procedure of using ten random starting points, specifically initializing a point by a normal distribution centered at zero, and then picking the best solution (the one with the lowest objective value) of ten runs.

We list the parameter and initial values we use in our implementations of SL/QP in Table 1.1 $0 < \gamma_1 < 1$, $\gamma_2 > 1$, $\Delta_m > 0$, $\delta_m > 0$, and $S \in \mathbb{N}$. We denote $k_{\max}$ the maximum number of iterations, $LSk_{\max}$ the maximum number of line-search steps, and η the backtracking contraction parameter.

Table 1.1: Control parameters and initial values required by algorithm 1

Parameter	Value	Parameter	Value	Parameter	Value
γ_1	`0.1`	S	`10`	$k_{\max}$	`20`
γ_2	`2.0`	N_0	N	$LSk_{\max}$	`20`
δ_m	`1.0e-r`	Δ_0	`1.0`	η	`0.5`

We analyze the performance on two time-delay systems described in [24]. The first example is a third-order feedback controller system of the form,

$$v'(t) = Av(t) + B(x)v(t-5),$$

with A and $B(x)$ defined to be,

$$A = \begin{pmatrix} -0.08 & -0.03 & 0.2 \\ 0.2 & -0.04 & -0.005 \\ -0.06 & -0.2 & -0.07 \end{pmatrix}$$

and

$$B(x) = \begin{pmatrix} -0.1 \\ -0.2 \\ 0.1 \end{pmatrix} \begin{pmatrix} x_1 & x_2 & x_3 \end{pmatrix}.$$

We present the sum of the results of the values and times in Table 1.2.

Table 1.2: Mean (standard deviation) for values and times for SLP and SQP (out of 500 sample runs). Value for each solver for each run is taken as the best of 10 random starting points. Time is the total clock time taken to perform the ten runs.

	value	time
SLP	-0.081 (0.053)	4.6 (1.1)
SQP	-0.088 (0.055)	5.3 (1.6)

The best value was found to be -0.239, at $x = (-0.21, 0.074, 1.38)$ and -0.129 at $x = (-0.036, 0.67, 0.94)$ for SLP and SQP respectively. For SLP, 16% of the initial random starting points corresponded to a stable (negative) value of the spectral abscissa, and 25% of the final iterations did, among all of the trials. For SQP these numbers were 10% and 23%, respectively.

The next example is given below,

$$
\begin{aligned}
&T_h\dot{x}_h(t) = -x_h(t-\eta_h) + K_b x_a(t-\tau_b) + K_u x_{h,set}(t-\tau_u),\\
&T_a\dot{x}_a(t) = -x_a(t) + x_c(t-\tau_e) + K_a(x_h(t) - \tfrac{1+q}{2}x_a(t) - \tfrac{1-q}{2}x_c(t-\tau_e)),\\
&T_d\dot{x}_d(t) = -x_d(t) + K_d x_a(t-\tau_d),\\
&T_c\dot{x}_c(t) = -x_c(t-\eta_c) + K_c x_d(t-\tau_c),\\
&\dot{x}_e(t) = -x_c(t) + x_{c,set}(t),
\end{aligned}
$$

with,

$$x_{h,set}(t) = \begin{pmatrix} K_1 & K_2 & K_3 & K_4 & K_5 \end{pmatrix} \begin{pmatrix} x_h(t) & x_a(t) & x_d(t) & x_c(t) & x_e(t) \end{pmatrix}^T.$$

The results comparing SLP and SQP, which are qualitatively similar as in the first example, are given in Table 1.3.

Table 1.3: Mean (standard deviation) for values and times for BFGS with and without an additional gradient sampling phase, SLP, and SQP (out of 500 sample runs).

	value	time
SLP	-0.083 (0.0062)	83.5 (140)
SQP	-0.088 (0.0103)	75.5 (76)

The best value was found to be -0.015, at $k = (2.2, -12, -7.5, -6.9, 0.35)$ and -0.016 at $k = (-0.77, -3.0, -3.6, -4.2, 1.4)$ for SLP and SQP respectively. For SLP, 2% of the initial random starting points corresponded to a stable (negative) value of the spectral abscissa, and 19% of the final iterations did, among all of the trials. For SQP these numbers were 5% and 11%, respectively.

It appears as though SQP and SLP perform similarly, both in terms of final objective value and time. In general it appears that, given enough random starting points, the algorithms are successful for a fair number of trials in obtaining a stable controller.

1.4 Conclusion

In this chapter we studied the eigenvalue optimization problem of minimizing the spectral abscissa for time delay systems. This problem is important for designing stabilizing controllers. We presented an algorithm that incorporated linear and quadratic models of eigenvalue surfaces corresponding to different eigenvalues in a sequential linear and sequential quadratic programming framework.

Our numerical results demonstrated the efficacy of the presented approach for minimizing the spectral abscissa of time-delay models, indicating that it appears to be competitive with the state of the art in terms of both finding a good local minimizer as well as in terms of computational speed. As such, the SL/QP algorithm is a promising approach for solving this class of problems.

1.5 Acknowledgements

This research was supported by Research Council KUL: PFV/10/002 Optimization in Engineering Center OPTEC, GOA/10/09 MaNet, Belgian Federal Science Policy Office: IUAP P7 (DYSCO, Dynamical systems, control and optimization, 2012-2017); ERC ST HIGHWIND (259 166). V. Kungurtsev was also supported by the European social fund within the framework of realizing the project "Support of inter-sectoral mobility and quality enhancement of research teams at the Czech Technical University in Prague", CZ.1.07/2.3.00/30.0034 and the Cisco-CTU Sponsored Research Agreement project WP5.

References

1. V. I. Arnold. On matrices depending on parameters. *Russian Mathematical Surveys*, 26(2):29–43, 1971.
2. J. V. Burke, A. S. Lewis, and M. L. Overton. Two numerical methods for optimizing matrix stability. *Linear Algebra and its Applications*, 351:117–145, 2002.
3. J. V. Burke, A. S. Lewis, and M. L. Overton. A robust gradient sampling algorithm for nonsmooth, nonconvex optimization. *SIAM Journal on Optimization*, 15(3):751–779, 2005.

4. J. V. Burke and M. L. Overton. Differential properties of the spectral abscissa and the spectral radius for analytic matrix-valued mappings. *Nonlinear Analysis: Theory, Methods & Applications*, 23(4):467–488, 1994.
5. J. V. Burke and M. L. Overton. Variational analysis of non-lipschitz spectral functions. *Mathematical Programming*, 90(2):317–351, 2001.
6. A. R. Conn, N. I. Gould, and P. L. Toint. *Trust region methods*, volume 1. Siam, 2000.
7. F. E. Curtis and X. Que. A quasi-newton algorithm for nonconvex, nonsmooth optimization with global convergence guarantees. Technical report, Lehigh, 2014.
8. T.-M.-T. Do and T. Artières. Regularized bundle methods for convex and non-convex risks. *The Journal of Machine Learning Research*, 13(1):3539–3583, 2012.
9. A. Fuduli, M. Gaudioso, and G. Giallombardo. Minimizing nonconvex nonsmooth functions via cutting planes and proximity control. *SIAM Journal on Optimization*, 14(3):743–756, 2004.
10. E. M. Gertz et al. *Combination trust-region line-search methods for unconstrained optimization*. PhD thesis, University of California, San Diego, 1999.
11. K. KIWIEL. Methods of descent for nondifferentiable optimization. *LECTURE NOTES IN MATHEMATICS*, 1133:1–360, 1985.
12. P. Lancaster. On eigenvalues of matrices dependent on a parameter. *Numerische Mathematik*, 6(1):377–387, 1964.
13. C. Lemaréchal. Chapter vii nondifferentiable optimization. *Handbooks in Operations Research and Management Science*, 1:529–572, 1989.
14. A. S. Lewis. Nonsmooth optimization and robust control. *Annual Reviews in Control*, 31(2):167–177, 2007.
15. A. S. Lewis and M. L. Overton. Nonsmooth optimization via quasi-newton methods. *Mathematical Programming*, 141(1-2):135–163, 2013.
16. J. R. Magnus. On differentiating eigenvalues and eigenvectors. *Econometric Theory*, 1(2):pp. 179–191, 1985.
17. W. Michiels and S.-I. Niculescu. *Stability and stabilization of time-delay systems: an eigenvalue-based approach*, volume 12. Siam, 2007.
18. D. Noll and P. Apkarian. Spectral bundle methods for non-convex maximum eigenvalue functions: first-order methods. *Mathematical programming*, 104(2-3):701–727, 2005.
19. F. Oustry. A second-order bundle method to minimize the maximum eigenvalue function. *Mathematical Programming*, 89(1):1–33, 2000.
20. M. L. Overton. On minimizing the maximum eigenvalue of a symmetric matrix. *SIAM Journal on Matrix Analysis and Applications*, 9(2):256–268, 1988.
21. M. L. Overton. Large-scale optimization of eigenvalues. *SIAM Journal on Optimization*, 2(1):88–120, 1992.
22. M. L. Overton and R. S. Womersley. On minimizing the special radius of a nonsymmetric matrix function: Optimality conditions and duality theory. *SIAM Journal on Matrix Analysis and Applications*, 9(4):473–498, 1988.
23. J. Vanbiervliet, B. Vandereycken, W. Michiels, S. Vandewalle, and M. Diehl. The smoothed spectral abscissa for robust stability optimization. *SIAM Journal on Optimization*, 20(1):156–171, 2009.
24. J. Vanbiervliet, K. Verheyden, W. Michiels, and S. Vandewalle. A nonsmooth optimisation approach for the stabilisation of time-delay systems. *ESAIM: Control, Optimisation and Calculus of Variations*, 14(03):478–493, 2008.
25. A. Zhigljavsky and A. Žilinskas. *Stochastic global optimization*, volume 9. Springer Science & Business Media, 2007.